



Projeto de um Banco de Dados Completo

Nessa aula vamos aplicar tudo que já aprendermos até então em um projeto completo de um cenário bastante comum: Um sistema de streaming de música, com assinatura. Criaremos a modelagem conceitual, lógica e os scripts para implementação no SGBD, além de inserções de dados arbitrários e algumas seleções.

DEFINIÇÃO DO ESCOPO E MODELAGEM CONCEITUAL

Para praticarmos tudo que já aprendemos até então vamos criar um projeto de um banco de dados completo (do levantamento de requisitos à modelagem física) de um sistema de streaming de música com assinatura usuários. Esta modelagem é parte de um cenário real e nesta sessão vamos definir o escopo do contexto, além de apresentar as etapas da modelagem conceitual.

Vamos começar pelo item principal do sistema: A música, que possui um título, uma duração, o ano em que foi gravada, a letra, o arquivo de áudio e um estilo (Jazz, Rock, Samba etc.). Cada música pertence à um artista e possui, também, uma série de TAGs associadas, que podem ser usadas para buscas (exemplo: “músicas dos anos 60”, “músicas de ninar” etc.). A música pertence à um artista. Cada artista possui um nome, uma bibliografia e um país de origem. Cada artista possui um ou mais álbuns e cada álbum possui um título, um ano de lançamento e uma gravadora. Lembre-se que o álbum é um container de músicas.



Com essa descrição já temos subsídios suficientes para criar a modelagem conceitual desta parte do projeto, que pode ser vista na figura 1. Compare cuidadosamente as cardinalidades de cada um dos relacionamentos, especialmente da música em relação ao álbum, que é dada por uma relação com entidade associativa incluindo-se o atributo "ordem" que define qual a ordenação da música em determinado álbum. É importante salientar que neste modelo a música pertence ao artista e o álbum, sendo um contêiner de músicas, relaciona-se única e exclusivamente com a música, pois imagine um álbum de vários artistas com músicas diferentes. Por isso, o álbum não é necessariamente do artista, nesta modelagem. De qualquer maneira, com um álbum, pode0se obter qual (ou quais) artistas compõe aquela coleção.

Após compreendido o sistema de armazenamento das músicas, podemos partir para o sistema dos usuários. Bom, o usuário possui um plano onde pode criar dois tipos de conta: Conta individual (valor X) ou conta família (valor X + 40%). Na conta família o usuário possui um ou mais dependentes, até 10. Lembre-se que cada dependente também é usuário do sistema. O plano possui uma vigência de renovação (30 dias, por exemplo) e é preciso armazenar o histórico de cobranças dos usuários (com os 4 últimos dígitos do cartão usado para cobrar). Os usuários ingressam no sistema por um e-mail e senha, mas podem colocar uma foto de perfil, seu nome ou apelido e dados para a cobrança automática no cartão.

Dito isso, podemos realizar a modelagem conceitual desta etapa, vista na figura 2. É muito importante entender bem essa parte pois há um processo de validação do pagamento que deverá ser feito pelo sistema baseado no modelo de dados que estamos apresentando.

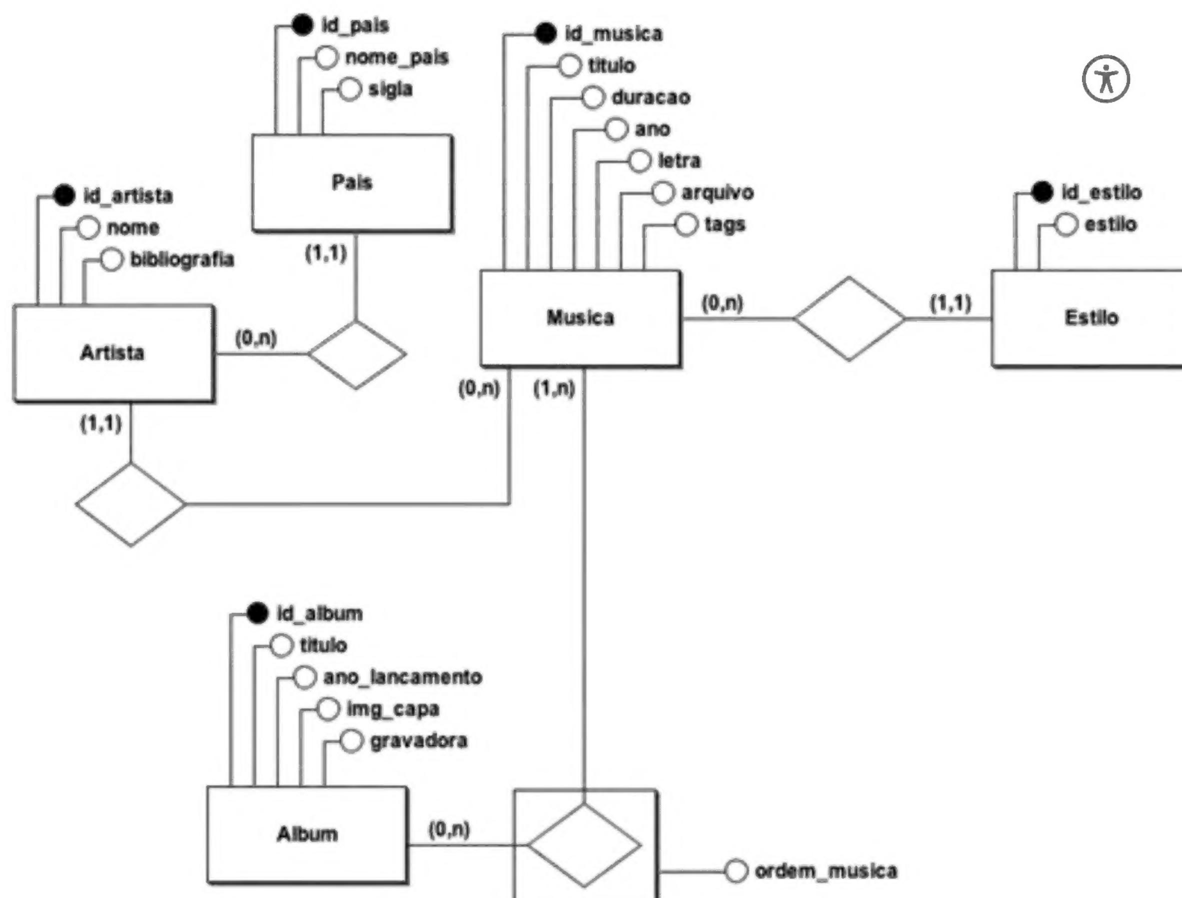


Figura 1 - Modelagem conceitual - parte 1

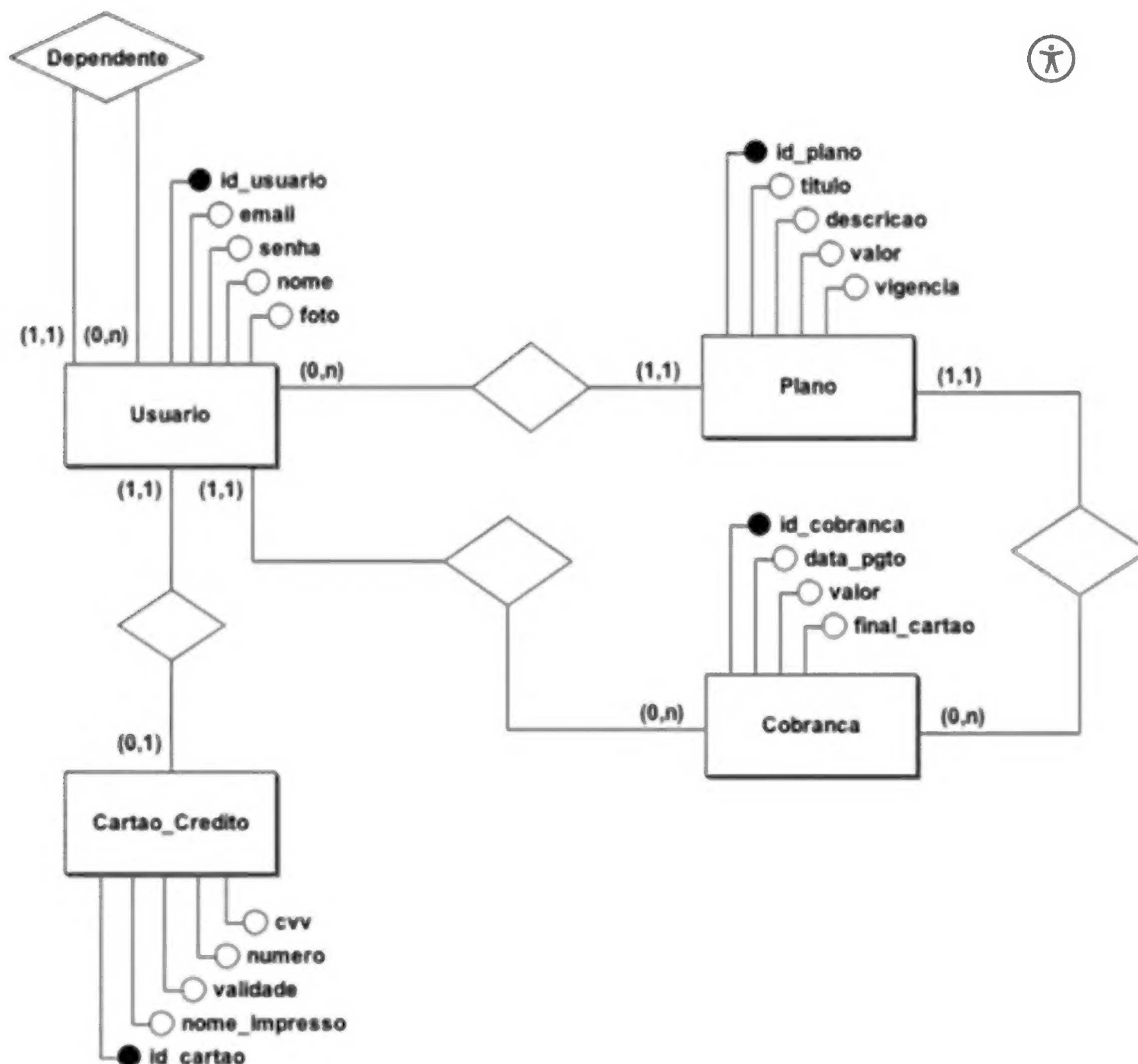


Figura 2 - Modelagem conceitual - parte 2

Finalmente os modelos se unem considerando que o usuário autenticado e com um plano válido (em vigência) pode ouvir músicas. Isso é feito pela programação, pois o banco de dados não implementa regras de negócio. Lembre-se que o plano é validado pela diferença de dias entre a data atual e última cobrança do usuário. Se estiver dentro da vigência do plano, ele está com uma conta ativa. Entretanto o usuário também pode criar suas próprias listas de reprodução, com quantas músicas ele quiser e as listas podem ter nomes e as músicas devem ter a ordenação que o usuário informar (que inclusive pode ser alterada posteriormente). Baseado nisso, a figura 3 mostra a união das duas partes da modelagem conceitual, através da associação do usuário com a música, pela lista de reprodução.

Outro ponto a ser observado é que neste sistema apenas o cartão de crédito é considerado meio de pagamento, para facilitar a modelagem que está suficientemente complexa para este exercício. Neste caso, após um pagamento ser efetuado no sistema (através do cartão de crédito do usuário), um novo registro deve ser incluído na entidade de cobrança. Este novo registro possui a data de pagamento e para validar se a conta do usuário está ativa, basta o software fazer a diferença de dias da data atual com a data do pagamento e subtrair da vigência de renovação do plano. Se o número for positivo, então a conta do usuário está ativa. Se for zero ou negativo, então uma nova cobrança deve ser gerada para aquele usuário. Lembre-se que usuários dependentes não possuem cobranças, apenas usuários responsáveis pela conta.

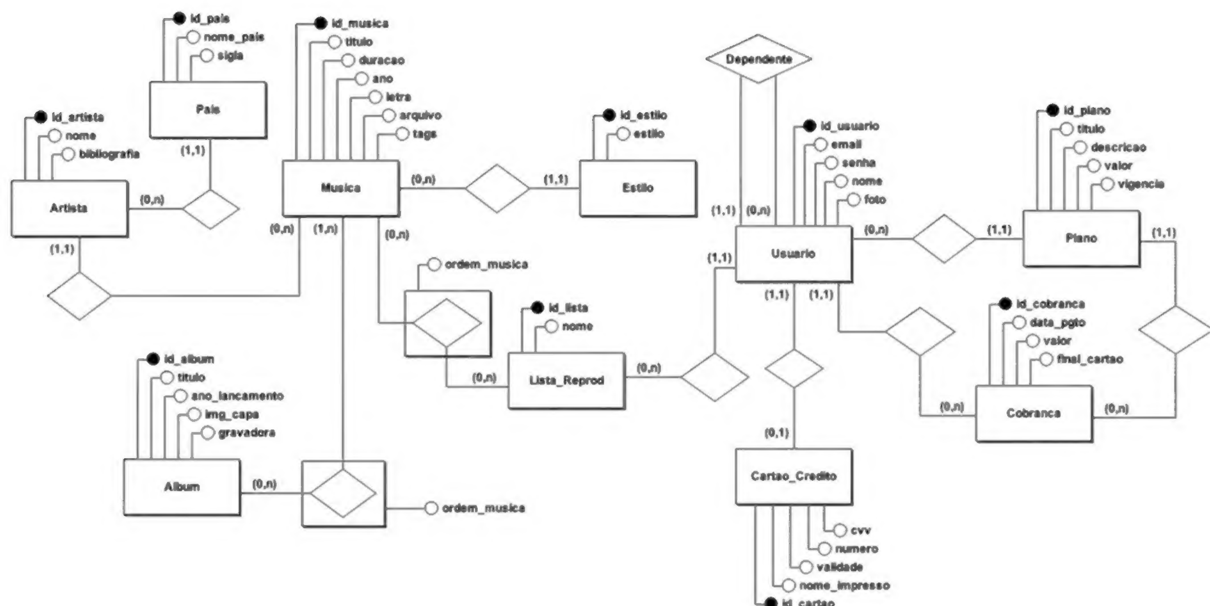


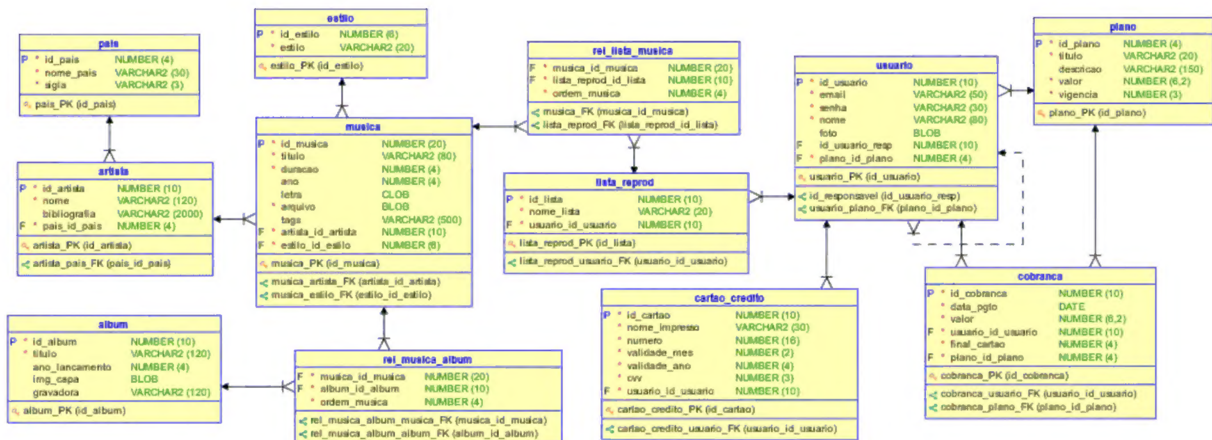
Figura 3 - Modelagem conceitual finalizada

DERIVAÇÃO PARA A MODELAGEM LÓGICA

Utilizando as regras de derivação da modelagem conceitual (fig. 3) para a modelagem lógica, temos como resultado o diagrama conforme mostrado na figura 4, desenvolvida através do *Oracle SQL Developer Data Modeler*. Note cuidadosamente que essa derivação gerou algumas tabelas de relacionamento novas, especialmente onde há, na modelagem conceitual, a

entidade associativa. Lembre-se que o atributo de uma entidade associativa se torna uma coluna na tabela de relacionamento como, por exemplo, o atributo que é capaz de identificar a ordem da música no álbum ou na lista de reprodução do usuário.

Repare também no autorelacionamento da tabela usuário que é um relacionamento não obrigatório. Cada conta de usuário, sendo uma conta familiar, poderá ter associada qual é a conta do usuário principal, que é de onde saem as cobranças do plano.



IMPLEMENTAÇÃO DA MODELAGEM FÍSICA

Após bem definidos os modelos conceituais e a modelagem lógica, podemos começar a executar a tão esperada modelagem física, gerando os scripts necessários para implementar este contexto, ou seja, os scripts da DDL do cenário em questão. O quadro abaixo mostra a criação das tabelas e respectivas restrições de integridade necessárias para implementação da modelagem lógica em questão.



```
--tabela país
CREATE TABLE pais (
    id_pais     NUMBER(4) NOT NULL,
    nome_pais   VARCHAR2(30) NOT NULL,
    sigla       VARCHAR2(3) NOT NULL,
    CONSTRAINT pais_pk PRIMARY KEY (id_pais)
);

--tabela estilo
CREATE TABLE estilo (
    id_estilo  NUMBER(6) NOT NULL,
    estilo     VARCHAR2(20) NOT NULL,
    CONSTRAINT estilo_pk PRIMARY KEY (id_estilo)
);

--tabela de albuns
CREATE TABLE album (
    id_album        NUMBER(10) NOT NULL,
    titulo          VARCHAR2(120) NOT NULL,
    ano_lancamento  NUMBER(4),
    img_capa        BLOB,
    gravadora       VARCHAR2(120),
    CONSTRAINT album_pk PRIMARY KEY (id_album)
);
```

```
--tabela de artistas:
CREATE TABLE artista (
    id_artista    NUMBER(10) NOT NULL,
    nome          VARCHAR2(120) NOT NULL,
    bibliografia  VARCHAR2(2000),
    pais_id_pais  NUMBER(4) NOT NULL,
    CONSTRAINT artista_pk PRIMARY KEY (id_artista),
    CONSTRAINT artista_pais_fk FOREIGN KEY (pais_id_pais)
        REFERENCES pais (id_pais)
);

--tabela de músicas:
CREATE TABLE musica (
    id_musica            NUMBER(20) NOT NULL,
    titulo               VARCHAR2(80) NOT NULL,
    duracao              NUMBER(4) NOT NULL,
    ano                  NUMBER(4),
    letra                CLOB,
    arquivo              BLOB NOT NULL,
    tags                 VARCHAR2(500),
    artista_id_artista   NUMBER(10) NOT NULL,
    estilo_id_estilo     NUMBER(6) NOT NULL,
    CONSTRAINT musica_pk PRIMARY KEY (id_musica),
    CONSTRAINT musica_artista_fk FOREIGN KEY (artista_id_artista)
        REFERENCES artista (id_artista),
    CONSTRAINT musica_estilo_fk FOREIGN KEY (estilo_id_estilo)
        REFERENCES estilo (id_estilo)
);

--tabela de relacionamento entre musica e album
CREATE TABLE rel_musica_album (
    musica_id_musica  NUMBER(20) NOT NULL,
    album_id_album    NUMBER(10) NOT NULL,
    CONSTRAINT rel_musica_album_album_fk FOREIGN KEY (album_id_album)
        REFERENCES album (id_album),
    CONSTRAINT rel_musica_album_musica_fk FOREIGN KEY (musica_id_musica)
        REFERENCES musica (id_musica)
);

--tabela do planos de assinaturas:
CREATE TABLE plano (
    id_plano   NUMBER(4) NOT NULL,
    titulo     VARCHAR2(20) NOT NULL,
    descricao  VARCHAR2(150),
    valor      NUMBER(6, 2) NOT NULL,
    vigencia   NUMBER(3) NOT NULL,
    CONSTRAINT plano_pk PRIMARY KEY (id_plano)
);

--tabela de usuários
CREATE TABLE usuario (
    id_usuario       NUMBER(10) NOT NULL,
    email            VARCHAR2(50) NOT NULL,
    senha            VARCHAR2(30) NOT NULL,
    nome             VARCHAR2(80) NOT NULL,
    foto             BLOB,
    id_usuario_resp  NUMBER(10),
    plano_id_plano   NUMBER(4) NOT NULL,
    CONSTRAINT usuario_pk PRIMARY KEY (id_usuario),
    CONSTRAINT id_responsavel FOREIGN KEY (id_usuario_resp)
        REFERENCES usuario (id_usuario),
    CONSTRAINT usuario_plano_fk FOREIGN KEY (plano_id_plano)
        REFERENCES plano (id_plano)
```

```
--tabela de cobranças:
CREATE TABLE cobranca (
    id_cobranca          NUMBER(10) NOT NULL,
    data_pgto            DATE NOT NULL,
    valor                NUMBER(6, 2) NOT NULL,
    usuario_id_usuario   NUMBER(10) NOT NULL,
    final_cartao         NUMBER(4) NOT NULL,
    plano_id_plano       NUMBER(4) NOT NULL,
    CONSTRAINT cobranca_pk PRIMARY KEY (id_cobranca),
    CONSTRAINT cobranca_plano_fk FOREIGN KEY (plano_id_plano)
        REFERENCES plano (id_plano),
    CONSTRAINT cobranca_usuario_fk FOREIGN KEY (usuario_id_usuario)
        REFERENCES usuario (id_usuario)
);

--tabela de cartões do usuário
CREATE TABLE cartao_credito (
    id_cartao            NUMBER(10) NOT NULL,
    nome_impresso        VARCHAR2(30) NOT NULL,
    numero               NUMBER(16) NOT NULL,
    validade_mes         NUMBER(2) NOT NULL,
    validade_ano         NUMBER(4) NOT NULL,
    cvv                  NUMBER(3) NOT NULL,
    usuario_id_usuario   NUMBER(10) NOT NULL,
    CONSTRAINT cartao_credito_pk PRIMARY KEY (id_cartao),
    CONSTRAINT cartao_credito_usuario_fk FOREIGN KEY (usuario_id_usuario)
        REFERENCES usuario (id_usuario)
);

--tabela da lista de reprodução
CREATE TABLE lista_reprod (
    id_lista             NUMBER(10) NOT NULL,
    nome_lista           VARCHAR2(20) NOT NULL,
    usuario_id_usuario   NUMBER(10) NOT NULL,
    CONSTRAINT lista_reprod_pk PRIMARY KEY (id_lista),
    CONSTRAINT lista_reprod_usuario_fk FOREIGN KEY (usuario_id_usuario)
        REFERENCES usuario (id_usuario)
);

--tabela que relaciona a lista com a música
CREATE TABLE rel_lista_musica (
    musica_id_musica        NUMBER(20) NOT NULL,
    lista_reprod_id_lista   NUMBER(10) NOT NULL,
    CONSTRAINT musica_fk FOREIGN KEY (musica_id_musica)
        REFERENCES musica (id_musica),
    CONSTRAINT lista_reprod_fk FOREIGN KEY (lista_reprod_id_lista)
        REFERENCES lista_reprod (id_lista)
```

MANIPULAÇÃO DOS DADOS

Após implementadas as tabelas no banco de dados, podemos começar a inserir alguns dados arbitrários para que possamos estudar as várias opções de seleção de dados e extração dos relatórios. Vamos iniciar agora, então, a parte da DML do banco de dados em questão, inserindo diversas músicas, artistas gêneros etc. e criando alguns usuários com seus respectivos dados de pagamentos. O quadro abaixo mostra a inserção de todos os dados necessários:



```
--criação dos países
INSERT INTO pais (id_pais, nome_pais, sigla)
VALUES (2, 'Reino Unido', 'UK');
INSERT INTO pais (id_pais, nome_pais, sigla)
VALUES (3, 'Finlandia', 'FL');
INSERT INTO pais (id_pais, nome_pais, sigla)
VALUES (4, 'Espanha', 'ES');
INSERT INTO pais (id_pais, nome_pais, sigla)
VALUES (5, 'Alemanha', 'AL');
INSERT INTO pais (id_pais, nome_pais, sigla)
VALUES (6, 'EUA', 'EUA');

--criação dos estilos
INSERT INTO estilo (id_estilo, estilo)
VALUES (1, 'Rock');
INSERT INTO estilo (id_estilo, estilo)
VALUES (2, 'Country');
INSERT INTO estilo (id_estilo, estilo)
VALUES (3, 'MPB');
INSERT INTO estilo (id_estilo, estilo)
VALUES (4, 'POP');
INSERT INTO estilo (id_estilo, estilo)
VALUES (5, 'Folk');
INSERT INTO estilo (id_estilo, estilo)
VALUES (6, 'Sertanejo');
INSERT INTO estilo (id_estilo, estilo)
VALUES (7, 'Blues');
INSERT INTO estilo (id_estilo, estilo)
VALUES (8, 'Jazz');

--criação dos albuns
INSERT INTO album (id_album, titulo, ano_lancamento, gravadora)
VALUES (1, 'Yellow Submarine', 1969, 'Abbey Road');
INSERT INTO album (id_album, titulo, ano_lancamento, gravadora)
VALUES (3, 'Help!', 1965, 'Parlophone');
INSERT INTO album (id_album, titulo, ano_lancamento, gravadora)
VALUES (5, 'Invitation', 1983, 'Warner Bros');

--criação dos artistas
INSERT INTO artista (id_artista, nome, bibliografia, pais_id_pais)
VALUES (1, 'The Beatles', 'Banda Inglesa muito famoso...', 2);
INSERT INTO artista (id_artista, nome, bibliografia, pais_id_pais)
VALUES (2, 'Jaco Pastorius', 'Baixista de Jazz...', 6);

--criação das músicas
INSERT INTO musica (id_musica, titulo, duracao, ano, letra, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (1, 'Yellow Submarine', 160, 1966, 'In the town..',
utl_raw.cast_to_raw('arquivo'), 'Anos 60;Psicolédica', 1, 1);

INSERT INTO musica (id_musica, titulo, duracao, ano, letra, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (2, 'Only a Northern Song', 214, 1969, 'If youre listening..',
utl_raw.cast_to_raw('arquivo'), 'Anos 60;Psicolédica', 1, 1);

INSERT INTO musica (id_musica, titulo, duracao, ano, letra, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (3, 'All Together Now', 131, 1966, 'One, two, three..',
utl_raw.cast_to_raw('arquivo'), 'Anos 60;Psicolédica', 1, 1);
```

```
INSERT INTO musica (id_musica, titulo, duracao, ano, letra, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (4, 'Hey Bulldog', 191, 1966, 'In the town...',
utl_raw.cast_to_raw('arquivo'), 'Anos 60;Psicolédica', 1, 1);

INSERT INTO musica (id_musica, titulo, duracao, ano, letra, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (5, 'Its All Too Much', 385, 1966, 'Its all too much...',
utl_raw.cast_to_raw('arquivo'), 'Anos 60;Psicolédica', 1, 1);

INSERT INTO musica (id_musica, titulo, duracao, ano, letra, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (6, 'All You Need Is Love', 191, 1966, 'Love, love, love...',
utl_raw.cast_to_raw('arquivo'), 'Anos 60;Psicolédica', 1, 1);

INSERT INTO musica (id_musica, titulo, duracao, ano, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (7, 'Invitation', 417, 1983, utl_raw.cast_to_raw('arquivo'), 'Jazz
famosos;Jazz improviso', 2, 8);

INSERT INTO musica (id_musica, titulo, duracao, ano, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (8, 'Amerika', 69, 1983, utl_raw.cast_to_raw('arquivo'), 'Jazz clássico;Jazz
improviso', 2, 8);

INSERT INTO musica (id_musica, titulo, duracao, ano, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (9, 'The Chicken', 409, 1983, utl_raw.cast_to_raw('arquivo'), 'Jazz
clássico;Jazz improviso', 2, 8);

INSERT INTO musica (id_musica, titulo, duracao, ano, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (10, 'Continuum', 268, 1983, utl_raw.cast_to_raw('arquivo'), 'Jazz
clássico;Jazz improviso', 2, 8);

INSERT INTO musica (id_musica, titulo, duracao, ano, arquivo, tags,
artista_id_artista, estilo_id_estilo)
VALUES (11, '"Liberty City', 273, 1983, utl_raw.cast_to_raw('arquivo'), 'Jazz
clássico;Jazz improviso', 2, 8);

--criação dos relacionamentos entre musica e album, com a ordem da música no album
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (1, 1, 1);
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (2, 1, 2);
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (3, 1, 3);
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (4, 1, 4);
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (5, 1, 5);
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (7, 5, 1);
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (8, 5, 2);
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (9, 5, 3);
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (10, 5, 4);
INSERT INTO rel_musica_album (musica_id_musica, album_id_album, ordem_musica)
VALUES (11, 5, 5);
```

```
--criação dos planos
INSERT INTO plano (id_plano, titulo, descricao, valor, vigencia)
VALUES(1, 'Individual', 'Plano individual total', 6.99, 30);
INSERT INTO plano (id_plano, titulo, descricao, valor, vigencia)
VALUES(2, 'Familiar', 'Plano familiar total', 9.79, 30);
--criação dos usuários
INSERT INTO usuario(id_usuario, email, senha, nome, plano_id_plano)
VALUES (1, 'madruguinha@descomplica.com.br', 'Senha123', 'Madruguinha', 2);
INSERT INTO usuario(id_usuario, email, senha, nome, id_usuario_resp, plano_id_plano)
VALUES (2, 'chiquinha@descomplica.com.br', 'Senha123', 'Chiquinha', 1, 2);
INSERT INTO usuario(id_usuario, email, senha, nome, plano_id_plano)
VALUES (3, 'chaves@descomplica.com.br', 'Senha123', 'Chavinho', 1);

--criação de algumas cobranças
INSERT INTO cobranca(id_cobranca, data_pgto, valor, usuario_id_usuario, final_cartao,
plano_id_plano)
VALUES(1, (TO_DATE('01/04/2020', 'dd/mm/yyyy')), 9.79, 1, 1234, 1);
INSERT INTO cobranca(id_cobranca, data_pgto, valor, usuario_id_usuario, final_cartao,
plano_id_plano)
VALUES(2, (TO_DATE('01/05/2020', 'dd/mm/yyyy')), 9.79, 1, 1234, 1);
INSERT INTO cobranca(id_cobranca, data_pgto, valor, usuario_id_usuario, final_cartao,
plano_id_plano)
VALUES(3, (TO_DATE('01/06/2020', 'dd/mm/yyyy')), 9.79, 1, 1234, 1);
INSERT INTO cobranca(id_cobranca, data_pgto, valor, usuario_id_usuario, final_cartao,
plano_id_plano)
VALUES(4, (TO_DATE('01/07/2020', 'dd/mm/yyyy')), 9.79, 1, 1234, 1);
INSERT INTO cobranca(id_cobranca, data_pgto, valor, usuario_id_usuario, final_cartao,
plano_id_plano)
VALUES(5, (TO_DATE('01/08/2020', 'dd/mm/yyyy')), 9.79, 1, 1234, 1);
INSERT INTO cobranca(id_cobranca, data_pgto, valor, usuario_id_usuario, final_cartao,
plano_id_plano)
VALUES(6, (TO_DATE('01/01/2020', 'dd/mm/yyyy')), 6.99, 3, 4321, 2);
INSERT INTO cobranca(id_cobranca, data_pgto, valor, usuario_id_usuario, final_cartao,
plano_id_plano)
VALUES(7, (TO_DATE('01/02/2020', 'dd/mm/yyyy')), 6.99, 3, 4321, 2);

--inserindo alguns cartões de crédito
INSERT INTO cartao_credito (id_cartao, nome_impresso, numero, validade_mes,
validade_ano, cvv, usuario_id_usuario)
VALUES(1, 'Roberto Bolaños', 1234432112344321, 10, 2030, 123, 3);
INSERT INTO cartao_credito (id_cartao, nome_impresso, numero, validade_mes,
validade_ano, cvv, usuario_id_usuario)
VALUES(2, 'Ramón Valdés', 4321123443211234, 12, 2022, 321, 1);
--criando algumas listas de reprodução
INSERT INTO lista_reprod(id_lista, nome_lista, usuario_id_usuario)
VALUES(1, 'Para curtir', 1);
INSERT INTO lista_reprod(id_lista, nome_lista, usuario_id_usuario)
VALUES(2, 'Para estudar', 1);
INSERT INTO lista_reprod(id_lista, nome_lista, usuario_id_usuario)
VALUES(3, 'Beatlemania', 2);
INSERT INTO lista_reprod(id_lista, nome_lista, usuario_id_usuario)
VALUES(4, 'Jazz legal', 3);
```

```
--inserdindo dados na relação da lista de rep com as músicas com a ordem da música na
lista
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_r
VALUES(1, 1, 1);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(2, 1, 2);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(4, 1, 3);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(6, 1, 4);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(11, 1, 5);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(1, 2, 3);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(5, 2, 2);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(1, 3, 1);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(2, 3, 1);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(3, 3, 2);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(4, 3, 3);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(5, 3, 5);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(6, 3, 4);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(8, 4, 1);
INSERT INTO rel_lista_musica (musica_id_musica, lista_reprod_id_lista, ordem_musica)
VALUES(11, 4, 2);
COMMIT;
```

```
SELECT
    musica.titulo,
    estilo.estilo
FROM
    musica
INNER JOIN
    estilo
ON musica.estilo_id_estilo = estilo.id_estilo
```

Resultado da Con...

SQL | Todas as Linhas Extraídas: 11 em 0,078 segundos

	TITULO	ESTILO
1	Yellow Submarine	Rock
2	Only a Northern Song	Rock
3	All Together Now	Rock
4	Hey Bulldog	Rock
5	Its All Too Much	Rock
6	All You Need Is Love	Rock
7	Amerika	Jazz
8	The Chicken	Jazz
9	Continuum	Jazz
10	"Liberty City	Jazz
11	Invitation	Jazz

Figura 4 - Seleção de músicas com seus respectivos estilos

Já a figura 5 mostra a execução de uma seleção um pouco mais refinada, onde são exibidas as músicas de um determinado álbum. Repare que INNER JOIN (que traz o que as tabelas possuem em comum na relação) é aplicado à mais de uma tabela simultaneamente.

```
SELECT
    musica.titulo AS titulo_musica,
    musica.duracao,
    artista.nome
FROM
    musica
    INNER JOIN rel_musica_album ON
            rel_musica_album.musica_id_musica = musica.id_musica
    INNER JOIN album ON rel_musica_album.album_id_album = album.id_album
    INNER JOIN artista ON musica.artista_id_artista = artista.id_artista
WHERE artista.nome = 'The Beatles'
      and album.titulo = 'Yellow Submarine'
ORDER BY rel_musica_album.ordem_musica;
```

Saída do Script × | Resultado da Consulta ×

SQL | Todas as Linhas Extraídas: 6 em 0,027 segundos

	TITULO_MUSICA	DURACAO	NOME
1	Yellow Submarine	160	The Beatles
2	Only a Northern Song	214	The Beatles
3	All Together Now	131	The Beatles
4	Hey Bulldog	191	The Beatles
5	Its All Too Much	385	The Beatles
6	All You Need Is Love	191	The Beatles

Figura 5 - Seleção das músicas de um determinado álbum

A figura 6 mostra a execução da extração das listas e suas respectivas músicas dos usuários. Repare nos critérios de ordenação, para organização do resultado final.

```sql
SELECT
    usuario.nome as usuario,
    lista_reprod.nome_lista,
    musica.titulo,
    musica.duracao,
    artista.nome AS artista
FROM
    lista_reprod
    INNER JOIN usuario ON lista_reprod.usuario_id_usuario = usuario.id_usuario
    INNER JOIN rel_lista_musica ON rel_lista_musica.musica_id_musica = lista_reprod.id_lista
    INNER JOIN musica ON rel_lista_musica.musica_id_musica = musica.id_musica
    INNER JOIN artista ON musica.artista_id_artista = artista.id_artista
ORDER BY  usuario.nome, lista_reprod.nome_lista, rel_lista_musica.ordem_musica;
```

Saída do Script × Resultado da Consulta ×

SQL | Todas as Linhas Extraídas: 8 em 0,032 segundos

	USUARIO	NOME_LISTA	TITULO	DURACAO	ARTISTA
1	Chavinho	Jazz legal	Hey Bulldog	191	The Beatles
2	Chavinho	Jazz legal	Hey Bulldog	191	The Beatles
3	Chiquinha	Beatlemania	All Together Now	131	The Beatles
4	Madruguinha	Para curtir	Yellow Submarine	160	The Beatles
5	Madruguinha	Para curtir	Yellow Submarine	160	The Beatles
6	Madruguinha	Para curtir	Yellow Submarine	160	The Beatles
7	Madruguinha	Para estudar	Only a Northern Song	214	The Beatles
8	Madruguinha	Para estudar	Only a Northern Song	214	The Beatles

Por fim, a figura 7 mostra o histórico de pagamentos de um determinado usuário (buscado pelo seu identificador único) e ordena os resultados pela data do pagamento.

```sql
SELECT
    cobranca.id_cobranca,
    cobranca.data_pgto,
    usuario.nome,
    cobranca.valor,
    cobranca.final_cartao,
    plano.titulo as plano
FROM
    cobranca
    INNER JOIN usuario ON cobranca.usuario_id_usuario = usuario.id_usuario
    INNER JOIN plano ON cobranca.plano_id_plano = plano.id_plano
WHERE usuario.id_usuario = 1
ORDER BY data_pgto;
```

Saída do Script × Resultado da Consulta ×

SQL | Todas as Linhas Extraídas: 5 em 0,012 segundos

	ID_COBRANCA	DATA_PGTO	NOME	VALOR	FINAL_CARTAO	PLANO
1	1	01/04/20	Madruguinha	6.99	1234	Individual
2	2	01/05/20	Madruguinha	9.79	1234	Familiar
3	3	01/06/20	Madruguinha	9.79	1234	Familiar
4	4	01/07/20	Madruguinha	9.79	1234	Familiar
5	5	01/08/20	Madruguinha	9.79	1234	Familiar

Figura 7 - Histórico de pagamentos de um determinado usuário

Atividade extra

Nome da atividade: Leia o texto sobre JOINs disponível no site abaixo e tente implementar o modelo descrito nos exemplos deste site além realizar as devidas seleções de dados.

SITE: https://www.techonthenet.com/oracle/joins.php

Se você precisar, ative o recurso de tradução do site em seu browser, para facilitar a leitura!

Referência Bibliográfica



DATE, C. J. Introdução a sistemas de banco de dados. Rio de Janeiro. Ed. Campus, 1991.



CHEN, Peter. Modelagem de dados: a abordagem entidade-relacionamento para projeto lógico. São Paulo: Makron Books, 1990.



MEDEIROS, L. F., Banco de dados, princípios e práticas, 1ª. ed., Ed. Intersaberes, 2013



PUGA, S., França E., GOYA M., Banco de dados: Implementação em SQL, PL/SQL e Oracle 11g, Ed. Pearson, 2013



ELMASRI R., NAVATHE, S., Sistemas de Banco de Dados, 4ªed., Ed. Pearson, 2005

Ir para exercício

